实验报告

张煜翔 520781910011

一、代码设计逻辑

声明与定义

A. 判断重定向、后台模式的结构体shellstatus_st

- 用于判断是否是前台执行的foreground；1为前台，0为后台。
- 用于判定是否需要输入输出重定向的参数infile和outfile
- 确定输出重定向的具体模式是覆盖还是加到末尾的outmode
- 确定shell路径的shellpath

B. 一些重要的函数接口以及外部环境变量

- 用于读取环境变量的environ
- 用于判断输入输出重定向并改变结构体shellstatus_st的函数check4redirection();
- 用于返回输出错误信息的errmsg()和返回系统错误信息的syserrmsg()
- 用于获取工作路径的getcwdstr()，这一函数在myshell中没有进行使用，但完成了//TODO部分
- 用于改变输出重定向写模式的redirection_op()
- 用于删除文件前路径的strippath()

主函数

A. 主函数中的声明

- 用于判断输入输出重定向读写模式的ostream和instream
- 用于存储command line和cwd的linebuffer和commandbuffer
- 用于存贮输入参数的args以及分别读取的指针arg
- prompt
- 用于判断输入输出重定向状态的sellstatus结构体status
- 存放readme文件的路径

B. 函数整体逻辑

1. 根据输入参数判断是进行command line还是batchfile

- 如果是command line，则直接进行之后的读取执行
- 如果是batchfile，则改变输入模式，在进入之后的对每一行分别执行
- 如果参数输入不对，则会返回用户错误信息

2. 获取环境路径

- 分别读取SHELL和readme的路径
- 将SHELL地址添加入环境变量

3. 通过信号防止**ctrl-C**与僵尸进程

4. 在**quit**指令之前对每一个**command line**进行执行

- 判断是否前台，并且通过输入模式选择是否要将prompt展示让用户输入还是直接执行batchfile
- 读取输入的指令，并存在args中
- 根据传入指令用check4redirection()确定输入输出重定向模式，随后根据具体指令进行执行，自己实现的会通过函数执行，否则会利用fork+exec传递给OS shell进行执行
  - 需要利用fork+exec执行的会通过execute函数进行，需要输入根据args以及status两个参数判断重定向执行模式
- 相应的指令与解释
  - cd
    - 通过new_dir存储要进入的新地址，如果不存在则返回当前地址；存在则将PWD变量设置为新地址
  - clr
    - 通过fork+exec实现，本质是在外部shell中运行clear指令
  - dir
    - 通过fork+exec实现，本质是在外部shell中运行ls -al指令
    - 还可以额外指定目录，均通过外部shell进行
  - echo
    - 首先阅读指令，根据check4direction判断是否需要进行重定向，需要则利用redirected_op切换输出模式
    - 随后依次读取，每一个tab或一个/多个space都会被压缩成一个space
    - 最终通过读取ostream来判断是直接输出还是进行文件流的输出
  - environ
    - 通过遍历环境变量environ来依次获取环境变量
    - 加入输出重定向功能，与之前思路一致
  - help
    - 通过shell执行more readmepath指令来打印出help me的内容
  - pause
    - 通过getpass忽略除了enter之外的输入，实现在enter之后继续执行的效果
  - 其他指令 -均通过execute函数传递给OS shell进行执行

其他重要函数

1. void check4redirection(char * *args, shellstatus* sstatus)

- 通过输入指令确认输入输出重定向的模式以及是否需要后台执行，写入结构体sstatus中
- 逻辑结构
  - 首先给sstatus设置缺省值，即均为默认的stdin、stdout
  - 对一行指令的每一个参数进行读取
    - 若存在<，则<之后的参数均不会被当成指令执行；并且随后改变sstatus中输入重定向文件的地址

        - 若存在>或>>，则进行输出重定向，>和>>之后的参数均不会被当成指令执行，如果是>则将文件写模式改为新建，>>则改为在源文件后附加或新建
        - 若读到&，则将sstatus的foreground参数改为0

2. void execute(char **args, shellstatus sstatus)

- 通过fork+execvp将指令传递给OS shell进行执行
    - 创建子进程，失败则返回系统错误信息
        - 创建成功，则通过sstatus中存储的输入输出重定向模式确定判断文件流和std的切换，并通过freopen进行实现
        - 随后在当前进程的环境变量中设置PARENT环境变量，按照Project1说明文件，设置为了myshell的路径
        - 通过execvp进行执行
    - 在父进程中，根据前台与后台进行的标识判断waitpid的参数。后台则子进程没有直接退出就直接返回；前台则等待子进程结束之后再返回

3. FILE *redirected_op(shellstatus status)

- 根据是否存在写文件来确定ostream的状态，不为空则根据相应的文件地址以及写模式返回输出流

二、用户使用手册

1. 名称

myshell-利用C语言实现的具有基本功能的shell

2. 用法概要

- 交互式执行（利用command line）
    - ```
      ./myshell
      ```
    - 随后将展示当前目录以及==>，可以在==>之后输入相应指令
    - 可以通过输入**quit**退出myshell，也可通过ctrl-C进行退出
- 批处理执行
    - ```
      ./myshell batchfile
      ```
    - batchfile为每一行代表一个指令的文件

3. 用户指令

- cd
    - 用法概要
        - ```
          cd <文件目录>
          ```
    - cd指令能够目录的跳跃，可以跳转到指定目录
          ```

    - 输入参数<文件目录>
        - 如果没有指定文件目录，则会输出当前目录
        - 如果指定的文件目录不存在则会输出错误信息
        - 存在则会跳转到相应目录
    - *输入的目录需要为当前目录下的相对路径或是要进入当前目录之外的绝对路径*
- clr
    - 用法概要
        - ```
          clr
          ```
    - 通过输入clr能够清空屏幕上显示的内容
- dir
    - 用法概要
        - ```
          dir <文件目录>
          ```
    - 能够输出指定目录的所有内容
    - *输入的目录需要为当前目录下的相对路径或是要指定当前目录之外的绝对路径*
- environ
    - 用法概要
        - ```
          environ
          ```
    - 将展示所有的环境变量
    - 程序的环境变量是指操作系统运行环境的一些参数，比如临时文件夹的位置，当前的路径等。在指定了环境变量之后，操作系统在运行一些程序时可以直接通过环境变量进行寻找调用，能够更方便的利用
        - 比如shell的环境路径被设置为myshell的绝对路径
        - 进行多进程时，子进程中含有父进程的PARENT环境变量
- echo
    - 用法概要
        - ```
          echo <输出文本>
          ```
    - 输出相应的文本并换行，当输出文本中存在多个space或tab，会压缩至一个
    - 在使用Batchfile时，可以利用echo来在返回文件中添加相应的输出，对脚本文件运行的程度进行了解
- pause
    - 用法概要
        - ```
          pause -能够停止Shell执行，并且只有在用户按下“ENTER”键之后才会继续执行
          ```
- quit
    - 用法概要
        - ```
          quit
          ```
    - 能够直接退出shell
- help
    - 用法概要

    - help
  - 能够输出用户指南，其中包含一些基本的使用方法
- 其他常见指令与Bash Shell的指令一致，可参考Bash shell语法指南

4. 高级使用方法

i) 重定向

输入重定向

- 输入重定向指可以将文件作为输入的参数传递给要执行的指令
- 用法

  > 命令 < 输入文件
  >
  > - 其中输入文件中包含了命令需要读取的参数
  > - 如能够统计文件行数的wc指令，当运行wc -l <文件>时，会返回行数以及文件名；但执行wc -l < <文件>则只会展示行数 ##### 输出重定向

- 输出重定向可以将命令的输出结果展示在一个特定的文件中
- 用法

  - > 命令 > 输出文件
  - > 命令 >> 输出文件

- 其中，>能够将命令的输出转移到输出的文件中，如果输出文件不存在则会新建；若存在则会覆盖
- >> 则是输出文件不存在则进行新建，存在则在文件之后添加新命令的输出结果
- 比如，执行echo nihao > nihao.txt，nihao.txt文件中则会有nihao；再执行echo nihao2 >> nihao.txt，nihao.txt文件中则会有nihao、nihao2(有换行) #### ii) 后台执行
- 后台执行可以针对一些输出过长或时间较长的任务。在远程连接执行时间较长的任务时，如果连接异常断开可能造成任务中断。避免这一现象的操作之一就是利用&进行后台执行
- 用法

  - > 命令 &

- 其代表Shell不需要等待该指令完成之后才能返回，而是只要该指令没有立即出错就会返回